UTM UNIVERSITI TEKNOLOGI MALAYSIA
INTERNATIONAL JOURNAL OF INNOVATIVE COMPUTING
ISSN 2180-4370
Journal Homepage : https://ijic.utm.my/

A Comparison of Transforming the User Stories and Functional Requirements into UML Use Case Diagram

Md Monirul Islam Molla[1], Johanna Ahmad[2] & Wan Mohd Nasir Wan Kadir[3]
Faculty of Computing
Universiti Teknologi Malaysia,
81310 UTM Johor Bahru,
Johor, Malaysia
Email: monirul.islam2000@graduate.utm.my[1]; johanna@utm.my[2]; wnasir@utm.my[3]

Submitted: 17/1/2024. Revised edition: 28/3/2024. Accepted: 12/5/2024. Published online: 31/5/2024
DOI: https://doi.org/10.11113/ijic.v14n1.463

***Abstract*—Software development life cycle is a continuous process for every developer community, including establishing the user requirements and system design until software maintenance. The system design phase contains the transformation of all-natural language requirements into models. The Unified Modelling Language (UML) use case model is regarded as one of the most helpful diagrams for a software developer to understand the higher version of the software blueprint. It's very useful to reflects the business requirements for any proposed software system. Manual analysis requires significant time and effort, highlighting the need for automated assistance. This study aims to enhance the machine learning techniques that have been utilized to transform user stories and functional requirements into a UML use case diagram. This research also investigate the use of an automated Natural Language Processing (NLP) tool and framework for mapping use case diagram elements from user stories and functional requirements written in natural languages. The proposed approach using a NLP based machine learning framework called BERT to predict and extract actors, use cases, and the relationship between actors and use cases. This approach has been applicable for both user stories and functional requirements. A dataset has been pre-processed in order to train this model and collected from various sources like Kaggle, software projects. There are 11 user stories and 11 functional requirements were selected to test the proposed solution. The data set from benchmark repositories. The extracted results during the whole NLP process are then mapped into actors, use cases, and relationships. This study was evaluated using both accuracy and precision values for performance measures. By applying NLP technologies, this research offers complete support for extracting elements and generating a use case diagram and revealed the 88% accuracy and 88% precision.**

***Keywords*—Software Requirements, UML diagram, NLP, BERT, Dataset, Accuracy and Precision value**

I. INTRODUCTION

In the field of Software Engineering, during Software Development Life Cycle (SDLC), Unified Modelling Language (UML) models are commonly used to blueprint a software system [1]. To clearly illustrate the features of developing software systems and minimize the ambiguity between the requirement specifications and the design, UML models are widely used in the software design phase. By involving users, a software system's blueprint enables developers to avoid misunderstandings regarding software development [2]. Before developing any software, system requirements engineering is the preliminary step to collecting all the business requirements from the stakeholders [3]. Following the collection of all business requirements, the developer should assess the textual requirements and produce an abstraction known as a requirement model.

To analyze a large number of textual requirements and extract the system features from the human language, currently, natural language processing (NLP) is the most suitable technique for developers. According to [4], "Techniques for natural language processing (NLP) may be advantageous for enhancing the quality of user stories [5]. ". NLP is a branch of artificial intelligence (AI) and computational linguistics that can be considered as a medium that facilitates human language interaction with computers or machines.

User stories and functional requirements are frequently used in software development as artifacts to document user requirements [6] but challenging to transform these user stories and functional requirements into UML use case diagrams [7-9]. Software requirements specifications are usually gathered in the requirements elicitation phases, often expressed in a natural language format. And user stories are derived from the requirements, which is a high-level description from the user's Perspective [10]. During the Analysis Phase of a Software Engineering Project, quickly capturing system features and user stories is considered a tool for rapid requirements analysis. The objective of this research is to compare whether the accuracy & precision value for the process of transforming user stories to use case diagrams is higher than the accuracy & precision value to transform use case diagrams from the requirements only. The research aimed to improve the process of transforming the user stories and functional requirements into a UML use case diagram by using this proposed approach [11].

This research will be focused on Natural Language Processing (NLP) [12], one of the branches of Artificial Intelligence (AI) that enables machines to understand the textual requirements based on user stories and functional requirements. The main scope of this research is to compare the accuracy value of the NLP techniques in transforming the user stories and functional requirements into a use case diagram. The transformation is vital in order to improve the understanding and communication between the enterprise, Product Owner, developers, and testers, and to considerably decrease the time and efficiency of the automatic generation of user stories and functional requirements into UML use case diagrams [13]. For this research, two case studies have been chosen from the Final Project Mobile Application Programming Course.

II. RELATED WORKS

The automation of the UML model (Use Case Diagram) from user requirements (User Stories & Functional Requirements) by using the technique of NLP is a comparatively new field. This chapter summarizes some of the most current research works that relied on using natural language processing to extract UML models from the textual specifications by using NLP and other technologies [14].

User stories refer to a quick, semi-formal statement that explains the requirements based on the user's point of view [4]. It can be used to get to know about the product and the client's desire within a short time. A user story illustrates functionality that will be valuable to either a user or consumer of a software system. User stories are a typical framework for conveying users' requirements in agile development initiatives [3,15]. One study state that "The three components of a user story are who, what, and why. The aspect of ''who'' refers to the system user or actor, ''what'' refers to the actor's desire, and ''why'' refers to the reason" [5]. After combining these aspects in one sentence with a specific structure: As a <aspect of who>, I want <aspect of what>, so that <aspect of why>.

According to a study, one of the most crucial processes in the software development process is requirements gathering [16]. The study suggested that throughout this analysis phase, all the requirements should be documented correctly because during the transformation process of requirements to use a case diagram, fixing the errors was quite time-consuming. The reason behind automating the UML model is to make the software development process smoothly. Both user stories and functional requirements are expressed in natural language, so researchers usually use "NL requirements".

NLP is often known as an automated (or semi-automated) system, allowing machines to understand natural human language. NLP is a diverse field of study that is closely linked to linguistics (Natural Language Processing Lecture Synopsis, n.d.). Usually, all the user stories and functional requirements are documented or expressed in natural language. Natural language processing (NLP) techniques are used to analyze these criteria, allowing for linguistic analysis and providing automated support [17]. Elallaoui also describes that Natural Language Processing (NLP) and Software Engineering (SE) are considered subfields of computer science. The Requirement Document and the Software Design Specification are examples of textual artifacts used throughout the analysis and design processes. According to the systematic literature review, researchers express the importance of using NLP in Requirement Engineering (RE) because "requirement specifications are developed in collaboration between software house analysts, users, and customers, and if requirements are stated in formal language, customers will not sign contracts" [18].

A study has been done by researchers associated with NLP application in requirements engineering [19, 20]. Based on the systematic literature review, Nazir *et al.* intensely focused on producing NLP applications for designing software requirements [19], whereas NL requirements were extracted for re-use in software development engineering [20].

There has recently been a lot of interest in automating software engineering tasks. One study surveyed to identify the most widely used modeling approaches in software engineering [9]. The authors stated that among all the unified modeling language (UML) diagrams, use case diagrams, class diagrams, and user stories are commonly used to define the user's goal. They also discussed the compatibility issues between user stories and use cases. Another author addressed the model transformation from Computational Independent Model (CIM) model into a Platform Independent Model (PIM), where the CIM model is specified by the Use case diagram [17].

Nowadays, the majority of researchers use natural language processing (NLP) and domain ontology to generate case diagrams from user stories. However, generating an image diagram as the final result of any enhancements or changes isn't allowed, which is a minor drawback [21]. In 2013 one study discussed the various Natural Language Processing and Text Retrieval strategies utilized in Software Engineering projects; moreover, the application fields of NLP in software engineering projects were described [22]. To extract the essential elements of use case diagrams from user stories in terms of producing UML use case diagrams, an automated textual requirements processing is provided by NLP. Some of the researchers discovered a lot of tools for analyzing a wide range of languages. Their system is flexible and has significant processing goals for several natural languages stated by the author [23].

Two different study described how natural language plays a significant role in requirements Engineering that has been recognized for a long period [5,24]. In 1981, a survey was conducted to provide a detailed summary of techniques for expressing requirements and specifications; authors stated, “the best language for requirements is natural language” [15].

In contrast, it is challenging to understand the requirements written in natural language by the software system developer because natural language requirements are ambiguous [25,26,27].

Thus, several semi-automated methodologies are available for software developers to construct a UML model from NL requirements utilizing NLP techniques [25,28]. Their techniques are mostly aligned with this research study, where the main purpose is to generate a use-case diagram from the NL requirements [29]. One latest study defined the most frequently used “NLP technologies are traditional analysis techniques (e.g., Parts of Speech tagging and text tokenization), general- purpose tools (e.g., Stanford CoreNLP and GATE) and generic language lexicons (WordNet and British National Corpus)”. Additionally, in recent years, several NLP techniques, toolkits, and algorithms have been applied to natural language requirements and transformed into UML use case diagrams [7].

III. RESEARCH METHODOLOGY

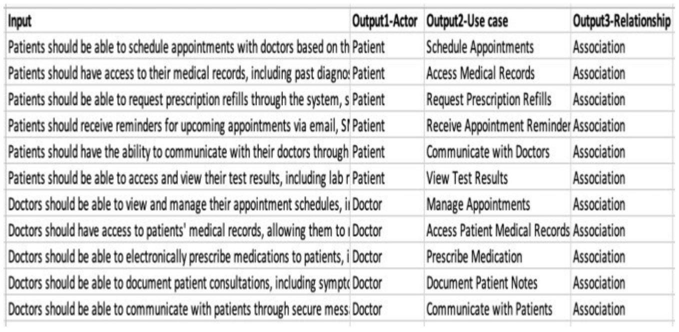
The methodology framework of this research consists of several phases which are (1) Literature Review, (2) Problem Definition (3) Experiment & Evaluation, and (4) Result Discussion & Conclusion. This section briefly explains the data acquiring sources and applicable data for this research.

For the NLP and ML models that are needed Data will be gathered from persons and organisations involved in software development as well as through web sites and other sources [30]. Online resources can include different publications, white papers, and freely accessible digital archives. Expect to take texts from this category of resources that describe software, provide a broad overview of a desired programme, and, in particular, contain software requirements or requirement language from software requirements specification documents. For both user stories and functional requirements dataset will be collected through multiple software projects and online resources. Few resources can be found in keggle and detail of the data as stated below:

1. Software requirements Data Set (https://www.kaggle.com/iamsouvik/software- requirements-dataset)

After collecting the dataset from various resources, this study examines and preprocess these data with expected input and output label as shown in Tabke I:

TABLE I. DATASET PREPARATION



Input	Output1-Actor	Output2-Use case	Output3-Relationship
Patients should be able to schedule appointments with doctors based on th	Patient	Schedule Appointments	Association
Patients should have access to their medical records, including past diagno:	Patient	Access Medical Records	Association
Patients should be able to request prescription refills through the system, s	Patient	Request Prescription Refills	Association
Patients should receive reminders for upcoming appointments via email, SI	Patient	Receive Appointment Reminder	Association
Patients should have the ability to communicate with their doctors through	Patient	Communicate with Doctors	Association
Patients should be able to access and view their test results, including lab r	Patient	View Test Results	Association
Doctors should be able to view and manage their appointment schedules, ii	Doctor	Manage Appointments	Association
Doctors should have access to patients' medical records, allowing them to i	Doctor	Access Patient Medical Records	Association
Doctors should be able to electronically prescribe medications to patients, i	Doctor	Prescribe Medication	Association
Doctors should be able to document patient consultations, including sympt	Doctor	Document Patient Notes	Association
Doctors should be able to communicate with patients through secure mess	Doctor	Communicate with Patients	Association

To examine this study 11 user stories and 11 functional requirements have been chosen from undergraduate project. The purpose of this research is to utilize a Natural Language Processing (NLP) tool to increase the efficiency and performance to transform user stories and functional requirements into a UML use case diagram [7,30]. Additionally, it purposes to conduct a study that covers the automatic generating of use case diagrams from the textual requirements written in natural language [31]. So, the proposed NLP techniques extract the necessary information from the natural language and finally draw the UM use case diagram. Based on their usability, feature-richness, and dependability, the following tools were identified in this study.

The following operations are frequently used to improve the quality and relevance of text data for summarization or other activities during document preprocessing. Modern natural language processing (NLP) technology was used to create the BERT (Bidirectional Encoder Representations from Transformers). By utilising a transformer architecture, it is intended to capture the contextual relationships and meanings of words. BERT can learn rich representations of words and phrases since it has been pre-trained on a vast corpus of text data. Following these stages, the BERT model extracts use case diagram elements from user stories and functional requirements.

Firstly, this model will be useful for text preprocessing. To eliminate noise, tokenize the text, and normalise it for consistency, the user stories and functional requirements will be preprocessed.

Secondly, the BERT tokenizer is used to tokenize the preprocessed text. Taking into account word boundaries and special characters, it divides the text into smaller chunks known as tokens. BERT can comprehend the contextual meaning of each token thanks to this tokenization.

Thirdly, Contextualised word embeddings are created by running the tokenized text through the BERT model.

Fourthly, to recognise and categorise named entities, such as actors and use cases, the BERT model may be adjusted for NER. The model gains accurate recognition and labeling skills by practicing on NER datasets annotated for the domain.

Fifthly, the BERT model may find connections between actors and use cases by examining the contextual embeddings of various things. Depending on the situation, it can learn to recognise patterns and distinguish relationship labels like "performs," "interacts with," or "depends on."

For creating UML usecase diagrams, a wide variety of CASE tools are available. After successfully extracting actors, use cases, and use case relationships, this system generates a use case diagram in the form of a PlantUML diagram.

Instead of utilising other tools, this project will make use of an integrated PlantUML solution since it is simple to build language code after deriving the use case diagram parts (PlantUML, n.d.). PlantUML makes it easier to create and share UML diagrams, which makes it a useful tool for discussing and visualising system designs

II. RESEARCH DESIGN AND IMPLEMENTATION

Having discussed the problem definition and research methodology of our proposed research in previous chapter, this chapter will concentrate on the proposed system's implementation process using various NLP and ML-related methods and libraries.

It begins with a discussion on how to identify Use case related elements using NLP and ML techniques for the prototype's implementation. This is followed by a detailed explanation of the implementation process, problems encountered, and proposed solutions upon [32].

A. Proposed Experimantal Design

In the experimental setup, the user will be asked to input the user stories or functional requirements in English text written in a natural language connected with business requirements. After entering the text Natural language Processing rule will be applied for Tokenization, POS tagging [39,40], and stemming, and all of these processes will be done automatically in the final prototype of this research. After successfully applying NLP to the textual requirements, the next task is to extract Actors and use cases and identify the relationship between actors and use cases. And finally, a use-cased diagram will be constructed.

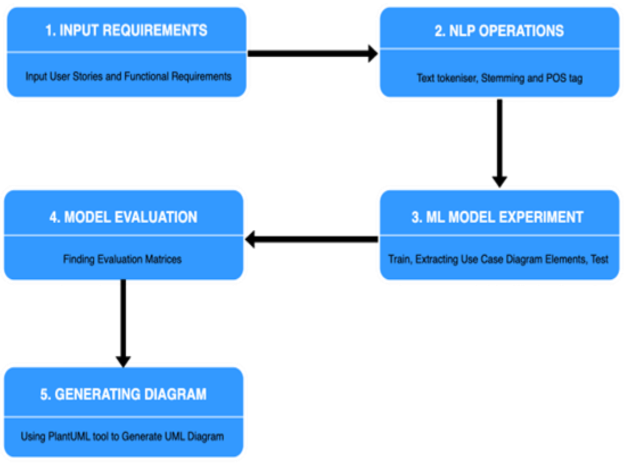

1. INPUT REQUIREMENTS
Input User Stories and Functional Requirements
2. NLP OPERATIONS
Text tokeniser, Stemming and POS tag
3. ML MODEL EXPERIMENT
Train, Extracting Use Case Diagram Elements, Test
4. MODEL EVALUATION
Finding Evaluation Matrices
5. GENERATING DIAGRAM
Using PlantUML tool to Generate UML Diagram

Fig. 1 Proposed process flow experimental design

According to the proposed approach in Fig. 1 Users may upload a CSV file for model training or testing, a file containing user stories or functional requirements [20], or they can just copy and paste a requirement language within the experimental design flow.

Using the NLP module, the system extracts named entities and word fragments. The ML module will then identify and extract the use cases, actors, and associations in accordance with the applied ML techniques. The proposed system model was modified with numerous output levels based on the BERT Model for this ML execution.

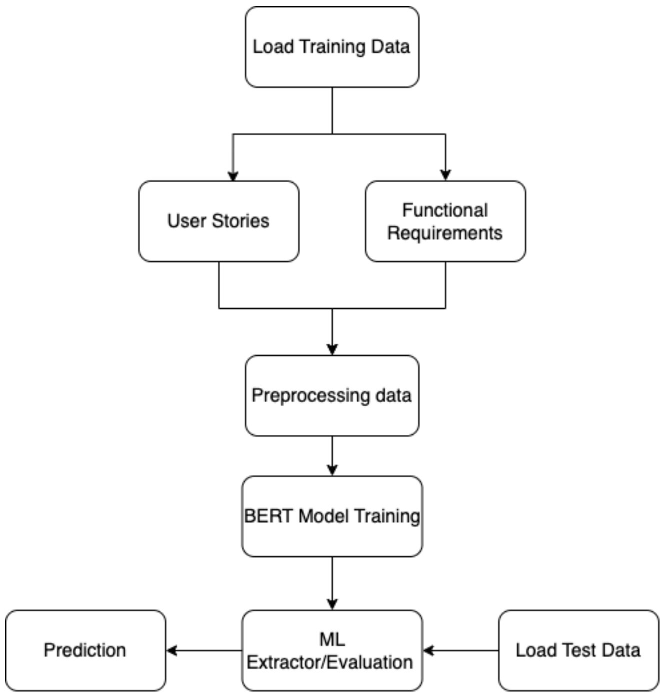


Fig. 2 The ML model training and evolution process flow

The study will then look at the evaluation matrices for this general comparison in transformation of user stories and functional requirements into use case diagram.

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)}$$

$$Precision = \frac{TP}{(TP+FP)}$$

The use case diagram is created using plant UML modelling. The user may edit and alter use cases, actors, classes, characteristics, and all kinds of connections in the system-generated diagrams. As a result, the output is extremely customisable, allowing the user to produce the required output in accordance with the given business needs.

B. Implementation

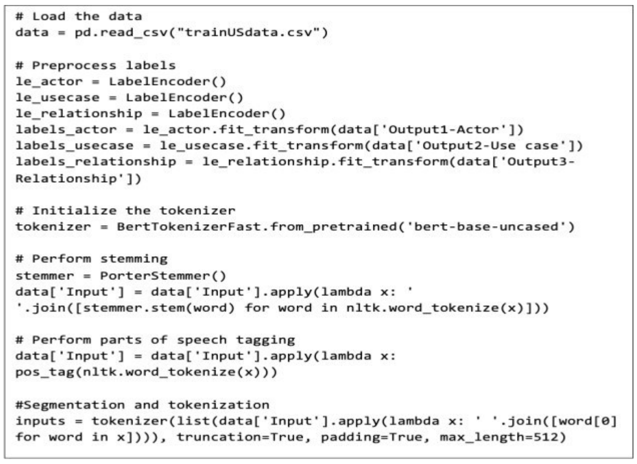
This section is a comprehensive examination of the above experiments. In this section, this paper will outline all the steps mentioned in Fig. 1 and the study will do step-by-step implementation from Input requirements to the UML use case diagram generation.

For NLP and ML implementation experimental approach for this examination will be carried out utilizing a Python- based experiment and suitable libraries and frameworks were chosen for the experiment based on the model and method we selected.

After compiling the original dataset for both User stories and Function requirements, we separated it into components based on the diagram elements. Then we begin working on model construction using each dataset that we have prepared.

And finally, this research will provide model for extracting use case diagram elements for the use case diagram generation. The solution will be examined by choosing only 11 user stories and 11 functional requirements from the two (2) undergraduate projects as a reference.

- **Input Requirements:** The first stage in the process flow for translating "user stories" into a UML use case diagram is to enter the requirements as text as shown in Fig. 1. This entails gathering and textually recording the user stories and functional requirements.
- **NLP Operations:** In this section, detail implementation of Natural Language Processing (NLP) operations was implemented starting from text tokenizing to Parts Of Speech (POS) tagging. The code snippet that follows uses the BERT model to perform many natural languages processing (NLP) operations mentioned: text tokenization, stemming, and POS tagging.



```
# Load the data
data = pd.read_csv("trainUSdata.csv")

# Preprocess labels
le_actor = LabelEncoder()
le_usecase = LabelEncoder()
le_relationship = LabelEncoder()
labels_actor = le_actor.fit_transform(data['Output1-Actor'])
labels_usecase = le_usecase.fit_transform(data['Output2-Use case'])
labels_relationship = le_relationship.fit_transform(data['Output3-
Relationship'])

# Initialize the tokenizer
tokenizer = BertTokenizerFast.from_pretrained('bert-base-uncased')

# Perform stemming
stemmer = PorterStemmer()
data['Input'] = data['Input'].apply(lambda x: '
'.join([stemmer.stem(word) for word in nltk.word_tokenize(x)]))

# Perform parts of speech tagging
data['Input'] = data['Input'].apply(lambda x:
pos_tag(nltk.word_tokenize(x)))

#Segmentation and tokenization
inputs = tokenizer(list(data['Input'].apply(lambda x: ' '.join([word[0]
for word in x]))), truncation=True, padding=True, max_length=512)
```

Fig. 3 Code snippet for NLP operations

- **ML Model Implementation:** This section contains the detail step to identifying Use case Diagram elements ML techniques.
    - **Define Dataset Class and Instantiate the Dataset:** In this code snippet, a dataset class is defined, and an instance of the dataset is instantiated.
    - **Define Class Model:** Instantiate an object of this class and train it on labelled data by defining the class model, allowing it to learn the patterns and relationships required to make correct predictions on fresh, unseen data.
    - **Instantiate The Model**
    - **Training Model:** This shows how to train a model with the specified dataset and model class.
    - **Model Inferencing:** Model inference is accomplished by the use of a function that accepts input data, performs the required preprocessing, feeds the data through the trained model, and outputs the projected results.
- **Model Evaluation Matrices [33]:** For the Accuracy and precision value assessment, this study has examined the user stories and functional requirements that have been chosen for two undergraduate projects to prepare test dataset.

TABLE II. EVALUATION METRICS

		Predicted Use Case Diagram elements	
		Yes	No
Actual Use Case Diagram elements	Yes	True Positive (TP)	False Negative (FN)
	No	False Positive (FP)	True Negative (TN)

- **Generating Use Case Diagram:** This the final stage for our proposed solution process flow for this research study. This section defines the methodology to automatically generate use case diagram by using PlantUML tool. With this implementation, we hope to create a class and use case diagram based on the components that the previous two implementations identified for both User Stories and Functional requirements. Aspects of the use case diagram elements that have been determined from the User stories or Functional requirements will be stored on a CSV file.

III. EVALUATION AND RESULTS

The implementation's outcomes are reported and evaluated in this section with regard to the study's goal of creating UML diagrams that correspond to Business Requirements. using machine learning and natural language processing. The primary comparisons in the evaluation are between the two sub-aims, which are the first to determine use case diagram elements by using NLP [34,35,36] and ML techniques from textual requirements like: User Stories and Functional Requirements and the second to build diagram using the diagram generating tool. A case study and solution from an undergraduate software project were used for evaluation along with the project's output.

A. Identifying use case diagram element from requirement text

Our proposed solution has able to extract use case diagram elements such as actors, use case and relationship between actor and use cases. The efficiency and dependability of the proposed approach are demonstrated by the comparison of accuracy and precision values in terms of extracting use case diagram elements from user stories and functional requirements.

TABLE III. EVALUATION MATRICES RESULT

	Identifying Actors		Identifying Use Cases		Identifying Relationship	
	User Stories	Functional Requirements	User Stories	Functional Requirements	User Stories	Functional Requirements
Accuracy	90%	100%	55%	64%	100%	100%
Precision	90%	100%	55%	64%	100%	100%

Based on above table, The higher accuracy and precision achieved in predicting actors from functional requirements (100%) compared to user stories (90%) suggest that the approach may be more suitable for functional requirements documents. This study achieved an accuracy of 55% and a precision of 55% in predicting the use cases from the given user stories. In Use case identification achieved an accuracy of 64% and a precision of 64% in predicting the use cases from the given functional requirements.

B. Result Comparison and Discussion

The study observes that the approach for identifying use case diagram elements performs slightly better when using functional requirements as input text requirements compared to user stories. The average accuracy and precision values are higher when utilizing functional requirements.

TABLE IV. AVERAGE ACCURACY AND PRECISION RESULT

	Identify Use Case Diagram Elements	
	User Stories	Functional Requirements
Average Accuracy	82%	88%
Average Precision	82%	88%

C. Generating Use Case Diagram

Once the relevant use case diagram elements are identified from the user stories and functional requirement text using NLP and ML models, this study can utilize the PlantUML tool to create the corresponding diagrams.

The diagrams can be generated by feeding the extracted information into PlantUML, which offers a text-based approach for diagram creation. This allows for the automatic generation of diagrams based on the identified elements, simplifying the process and eliminating the need for manual diagram drawing.

a) Usecase Diagram with User stories:

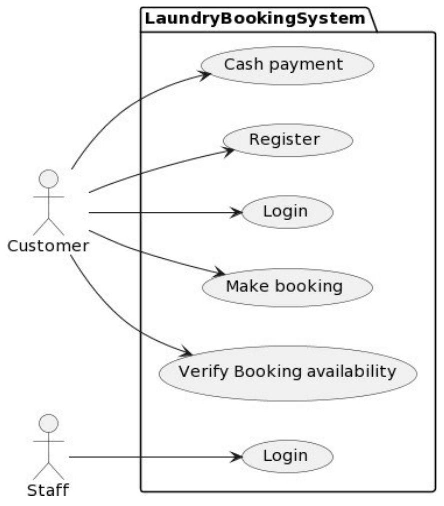


Fig. 4 The Generated Use case diagram from user stories

b) Usecase Diagram with Functional Requirements:

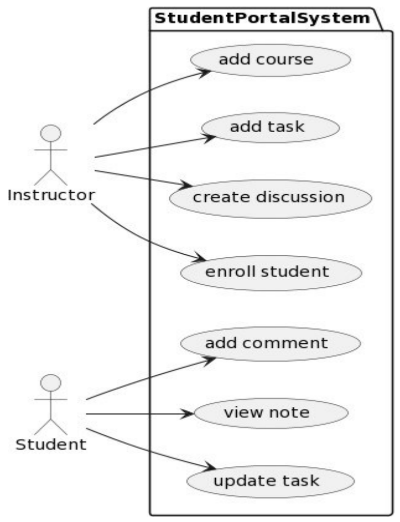


Fig. 5 The Generated Use case diagram from functional requirements

IV. ACHIEVEMENT AND FUTURE WORKS

Software developers can benefit greatly from the automatic creation of use case diagrams from natural language requirements by utilising natural language processing (NLP) technology. Through the use of machine learning techniques such as BERT, this method effectively converts functional requirements and user stories into UML use case models. It expedites the system design phase with minimal manual intervention, saving time and effort. This automated procedure speeds up work, makes it easier to comprehend business needs, and streamlines the software development lifecycle. Based on the result analysis this study can conclude that the proposed technique expresses a greater percentage of Accuracy 88% and Precision 88% value in terms of extracting and generating use case diagrams from user stories and functional requirements. Almost no research attempt for both user stories and functional requirements wherever in this study the proposed approach empirically evaluated both NL requirements. This suggested approach is able to identify the elements with precision, ensuring that the identified elements align closely with the actual elements present in the requirement

However, owing to the research's time constraints and the task's complexity, the author has designated it as a future development using the machine learning model, which requires a high degree of intelligence, high accuracy, and a large quantity of data for training. In this case, the upgraded version of a machine learning or deep learning model should be evaluated.

The current proposed approach has some drawbacks in terms of identifying complex types of relationships between actors and use cases like generalization and specialization (e.g. include and exclude). This research will not be able to deal with this complex relationship. These challenges will be addressed in future work to identify other types of relationships when generating use case diagrams.

ACKNOWLEDGMENT

We would like to thank Universiti Teknologi Malaysia for their full support for this research.

REFERENCES

[1] Alexander, I., Ian F., & Maiden, N. (2004). Scenarios, stories, use cases: Through the systems development life-cycle.

[2] Raharjana, I. K., Harris, F., & Justitia, A. (2020). Tool for generating behavior-driven development test-cases. *Journal of Information Systems Engineering and Business Intelligence*, *6*(1), 27-36. https://doi.org/10.20473/JISEBI.6.1.27-36.

[3] Behutiye, W., Karhapää, P., López, L., Burgués, X., Martínez-Fernández, S., Vollmer, A. M., Rodríguez, P., Franch, X., & Oivo, M. (2020). Management of quality requirements in agile and rapid software development: A systematic mapping study. *Information and Software Technology*, *123*(November 2019), 106225. https://doi.org/10.1016/j.infsof.2019.106225.

[4] User Stories Applied: For Agile Software Development-Mike Cohn-Google Books. (n.d.). Retrieved May 15, 2022, from https://books.google.com.my/books?hl=en&lr=&id=SvIwuX4SVigC&oi=fnd&pg=PR13&dq=M.+Cohn,+User+Stories+Applied:+for+Agile+Software+Development.+&ots=VrWaccwYMQ&sig=HSCD12M6PcRYTK1kxRREn5Ejlww&redir_esc=y#v=on epage&q=M. Cohn%2C User Stories Applied%3A for Agile Software Development.&f=false.

[5] Yalla, P., & Sharma, N. (2015). Integrating natural language processing and software engineering. *International Journal of Software Engineering and Its Applications, 9*(11), 127-136. https://doi.org/10.14257/IJSEIA.2015.9.11.12.

[6] Seresht, S. M., & Ormandjieva, O. (2008). Automated assistance for use cases elicitation from user requirements text. *11th Workshop on Requirements Engineering, WER 2008-Proceedings, October 2012*, 128-139.

[7] Ben Abdessalem Karaa, W., Ben Azzouz, Z., Singh, A., Dey, N., S. Ashour, A., & Ben Ghazala, H. (2016). Automatic builder of class diagram (ABCD): An application of UML generation from functional requirements. *Software: Practice and Experience*, *46*(11), 1443-1458. https://doi.org/10.1002/SPE.2384.

[8] Madanayake, R., Dias, G. K. A., & Kodikara, N. D. (2016). Use stories vs UML use cases in modular transformation. *International Journal of Scientific Engineering and Applied Science*, *3*(1), 2395-3470.

[9] Mining Multimedia Documents-Wahiba Ben Abdessalem Karaa, Nilanjan Dey-Google Books. (n.d.). Retrieved May 15, 2022. https://books.google.com.my/books?hl=en&lr=&id=-EQPEAAAQBAJ&oi=fnd&pg=PA67&dq=transform+functional+requirements+to+use+case+diagram+by+using+nlp+&ots=wFgXK2rNR9&sig=qdXrvBy03NtMRzYWMfFlSEse0E0&redir_e sc=y#v=onepage&q&f=false.

[10] Lucassen, G., Dalpiaz, F., Van Der Werf, J. M. E. M., & Brinkkemper, S. (2015). Forging high-quality user stories: towards a discipline for agile requirements. *2015 IEEE 23rd International Requirements Engineering Conference, RE 2015-Proceedings,* 126-135. https://doi.org/10.1109/RE.2015.7320415.

[11] Letsholo, K. J., Zhao, L., & Chioasca, E. V. (2013). TRAM: A tool for transforming textual requirements into analysis models. *28th IEEE/ACM International Conference on Automated Software Engineering, ASE 2013-Proceedings*, 738-741. https://doi.org/10.1109/ASE.2013.6693146.

[12] Zhao, L., Alhoshan, W., Ferrari, A., Letsholo, K. J., Ajagbe, M. A., Chioasca, E. V., & Batista-Navarro, R. T. (2021). Natural language processing for requirements engineering. *ACM Computing Surveys*, 54(3), 1-41. https://doi.org/10.1145/3444689.

[13] Maatuk, A. M., & Abdelnabi, E. A. (2021). Generating UML use case and activity diagrams using NLP techniques and heuristics rules. *ACM International Conference Proceeding Series*, 271-277. https://doi.org/10.1145/3460620.3460768.

[14] Alksasbeh, M. Z., Alqaralleh, B. A. Y., Alramadin, T. A., & Alemerien, K. A. (2017). An automated use case diagrams generator from natural language requirements. *Journal of Theoretical and Applied Information Technology, 95*(5), 1182-1190.

[15] Abbott, R. J., & Moorhead, D. K. (1981). Software requirements and specifications: A survey of needs and languages. *Journal of Systems and Software*, *2*(4), 297-316. https://doi.org/10.1016/0164-1212(81)90004-2.

[16] D. Joshi, S., & Deshpande, D. (2012). Textual requirement analysis for UML diagram extraction by using NLP. *International Journal of Computer Applications, 50*(8), 42-46. https://doi.org/10.5120/7795-0906.

[17] Houston, D. X., & Buettner, D. J. (2013). Modeling user story completion of an agile software process. *ACM International Conference Proceeding Series,* 88-97. https://doi.org/10.1145/2486046.2486063.

[18] Deeptimahanti, D. K., & Sanyal, R. (2009). An innovative approach for generating static UML models from natural language requirements. *Communications in Computer and Information Science, 30*, 147-163. https://doi.org/10.1007/978- 3-642-10242-4_13.

[19] Bakar, N. H., Kasirun, Z. M., & Salleh, N. (2015b). Feature extraction approaches from natural language requirements for reuse in software product lines: A systematic literature review. *Journal of Systems and Software, 106*, 132-149.

https://doi.org/10.1016/J.JSS.2015.05.006.

[20] Raharjana, I. K., Siahaan, D., & Fatichah, C. (2021). User stories and natural language processing: a systematic literature review. *IEEE Access, 9*, 53811-53826. https://doi.org/10.1109/ACCESS.2021.3070606.

[21] Overmyer, S. P., Lavoie, B., & Rambod, O. (2001). Conceptual modeling through linguistic analysis using LIDA. *Proceedings-International Conference on Software Engineering*, 2001-May, 401-410. https://doi.org/10.1109/ICSE.2001.919113.

[22] Kochbati, T., Li, S., Gérard, S., & Mraidha, C. (2021). From user stories to models: A machine learning empowered automation. *MODELSWARD 2021-Proceedings of the 9th International Conference on Model-Driven Engineering and Software Development, Modelsward*, 28-40. https://doi.org/10.5220/0010197800280040.

[23] Azzazi, A. (2017). A Framework using NLP to automatically convert user-stories into use cases in software projects. *17*(5), 71-76.

[24] Bakar, N. H., Kasirun, Z. M., & Salleh, N. (2015a). Feature extraction approaches from natural language requirements for reuse in software product lines: A systematic literature review. *Journal of Systems and Software*, *106*, 132-149. https://doi.org/10.1016/j.jss.2015.05.006.

[25] Yamout, F., Demachkieh, R., Hamdan, G., & Sabra, R. (n.d.). Further enhancement to the Porter's stemming algorithm.

[26] Deeptimahanti, D. K., & Babar, M. A. (2009). An automated tool for generating UML models from natural language requirements. *ASE2009-24th IEEE/ACM International Conference on Automated Software Engineering*, 680-682. https://doi.org/10.1109/ASE.2009.48.

[27] Elallaoui, M., Nafil, K., & Touahni, R. (2018). Automatic transformation of user stories into UML use case diagrams using NLP techniques. *Procedia Computer Science*, *130*, 42-49. https://doi.org/10.1016/j.procs.2018.04.010.

[28] Gulia, S., & Choudhury, T. (2016). An efficient automated design to generate UML diagram from Natural Language Specifications. *Proceedings of the 2016 6th International Conference-Cloud System and Big Data Engineering, Confluence 2016,* 641-648. https://doi.org/10.1109/CONFLUENCE.2016.7508197.

[29] Elallaoui, M., Nafil, K., & Touahni, R. (2015). Automatic generation of UML sequence diagrams from user stories in Scrum process. *2015 10th International Conference on Intelligent Systems: Theories and Applications, SITA 2015.* https://doi.org/10.1109/SITA.2015.7358415.

[30] Cayaba, C., Rodil, J. A., & Lim, N. R. (n.d.). CAUse: Computer automated use case diagram generator. *Analyzer,* 17-20.

[31] Dawood, O. S., & Sahraoui, A.-E.-K. (2017). From requirements engineering to UML using Natural Language Processing–Survey Study. *European Journal of Engineering Researchand Science*, *2*(1), 44. https://doi.org/10.24018/ejers.2017.2.1.236.

[32] Arman, N., & Jabbarin, S. (2015). Generating use case models from arabic user requirements in a semiautomated approach using a natural language processing tool. *Journal of Intelligent Systems, 24*(2), 277-286. https://doi.org/10.1515/JISYS-2014- 0092/MACHINEREADABLECITATION/RIS.

[33] Jilcha Sileyew, K. (2020). Research design and methodology. cyberspace. https://doi.org/10.5772/INTECHOPEN.85731.

[34] Markoulidakis, I., Rallis, I., Georgoulas, I., Kopsiaftis, G., Doulamis, A., & Doulamis, N. (2021). Multiclass confusion matrix reduction method and its application on net promoter score classification problem. *Technologies 2021, 9*(4), 81. https://doi.org/10.3390/TECHNOLOGIES9040081.

[35] Narawita, C. R., & Vidanage, K. (2018). UML generator–use case and class diagram generation from text requirements. *International Journal on Advances in ICT for Emerging Regions (ICTer), 10*(1), 1. https://doi.org/10.4038/icter.v10i1.7182.

[36] Natural Language Processing Lecture Synopsis. (n.d.). Retrieved May 15, 2022, http://www.cl.cam.ac.uk/users/aac/ Nazir, F., Butt, W. H., Anwar, M. W., & Khan Khattak, M. A. (2017). The Applications of Natural Language Processing (NLP) for Software Requirement Engineering - A Systematic Literature Review. Lecture Notes in Electrical Engineering, 424, 485-493. https://doi.org/10.1007/978-981-10-4154-9_56

[37] Ryan, K. (1993). The role of natural language in requirements engineering. *Proceedings of the IEEE International Conference on Requirements Engineering*, February 1970, 240-242. https://doi.org/10.1109/ISRE.1993.324852.

.